(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 003 115 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
24.05.2000 Bulletin 2000/21

(51) Int. Cl.[7]: **G06F 17/60**

(21) Application number: **99108757.8**

(22) Date of filing: **03.05.1999**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **03.09.1998 US 146711**

(71) Applicant:
Hewlett-Packard Company
Palo Alto, California 94304 (US)

(72) Inventors:
- **Gile, Ronald R.**
  Fort Collins, CO 80528 (US)
- **Makinen, Bruce A.**
  Fort Collins, CO 80525 (US)

(74) Representative:
Schoppe, Fritz, Dipl.-Ing.
Schoppe, Zimmermann & Stöckeler
Patentanwälte
Postfach 71 08 67
81458 München (DE)

(54) Audio/video from internet direct to compact disc through web browser

(57) A system and method for automatically creating user-customized compact discs (CDs) (112) containing multimedia tracks available over the Internet is presented. CD creation software (106) resident on the user's computer (102) or that is downloaded (304) from a web site that contains managed multimedia content is launched (308) from an Internet browser (104). The CD creation software (106) allows the user to select desired tracks (204) from a list of multimedia tracks (202) available from the managed web site. The selected tracks (204) are downloaded (318) from the web site and recorded onto a portable CD (112) at the user's request (212), or automatically per a user-specified schedule (214).

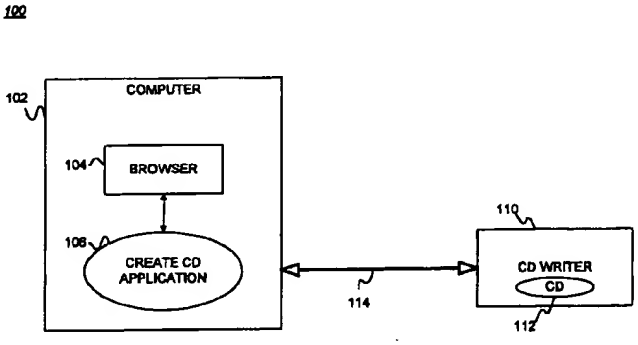

FIG. 1

EP 1 003 115 A2

Description

Field of the Invention

[0001] The present invention pertains generally to Internet services, and more particularly, to a system and method for allowing a user to select and download audio/visual tracks from the Internet and to record them directly onto a CD via a web browser for later retrieval.

Background of the Invention

[0002] The proliferation of the World Wide Web through the Internet has made available a wealth of information with nearly instantaneous access time. Much of the information available is in multimedia (i.e., audio and/or video) format, including music tracks and news reports. The multimedia content may be completely dynamic, being updated daily, hourly, or even broadcast live.

[0003] It is often convenient to capture a selection of various tracks available over the Internet on a local portable media for later playback. With the availability now of writable compact discs (CDs) and the massive user base of CD players already in use, it would be desirable to provide a system and method for allowing a user to select and download various audio/visual tracks from the Internet and to record them directly onto a CD for, later retrieval.

[0004] Depending on the size and format of the multimedia track, the bandwidth of the user's Internet connection, and the amount of traffic on the Internet at the time of download, the process required in visiting a service site, downloading a requested track, and listening to or watching the downloaded track may be time-consuming and/or inconvenient.

[0005] Accordingly, a need also exists for a system and method for selecting audio/video tracks from Internet multimedia service providers, automatically downloading the selected tracks, and writing the downloaded tracks to a CD for later retrieval by the user. It would also be convenient to provide a method for setting up a profile specifying the tracks and time desired, and having the CD created automatically without user intervention.

Summary of the Invention

[0006] The present invention provides a system and method for allowing users, via an Internet browser, to select various audio/visual tracks from Internet multimedia service providers, and have them downloaded and written to a CD using a CD writer. The CD may then be played at the user's convenience using, depending on the particular media, a CD or DVD drive. The invention allows the user to create a CD containing only the sounds, images, and information desired. In addition, the CD may be scheduled to be created without any user intervention to allow the CD to be available at the user's convenience.

Brief Description of the Drawing

[0007] The invention will be better understood from a reading of the following detailed description taken in conjunction with the drawing in which like reference designators are used to designate like elements, and in which:

FIG. 1 is a block diagram of a system in accordance with the invention;
FIG. 2 is an illustrative embodiment of a user interface in accordance with the invention; and
FIG. 3 is a flowchart of a method of the invention.

Detailed Description

[0008] FIG. 1 is a block diagram of a system 100 in accordance with the invention. In system 100, a computer 102 running a create CD application 106 in accordance with the invention via an Internet browser 104 downloads user selected audio/video tracks from the World Wide Web 120 over an Internet connection 122. The user selects tracks from a list of available services, displayed by a user interface of application 106, in any desired combination and order. Application 106 schedules the selected downloaded tracks for recording to CD 112 in CD writer 110. At the scheduled time, the selected downloaded tracks are written to CD 112 via a communication interface 114. Once the CD 112 is recorded, the user may play back the contents of the CD on any CD drive compatible with the particular medium of CD 112. To initially obtain the create CD application 106, a user navigates, via the user's Internet browser 104, to a web site that allows the user to download the create CD application 106 and downloads the application. When the user launches a CD write operation from browser 104, a user interface is presented that allows the user to select audio/video tracks from multimedia service providers.

[0009] FIG. 2 is an illustrative embodiment of a user interface 200 displayed within the users Internet browser 104 in accordance with the invention. As shown, user interface 200 includes a list of available services 202 and a list of selected services 204. The list of available services 202 includes a list of audio/video tracks made available by service providers. The list of available services 202 is typically configured by the provider of the user interface 200. Accordingly, the user interface 200 may vary from provider to provider, along with the services available through their user interface 200.

[0010] User interface 200 includes adding means 206 for allowing the user to select tracks from the list of available services 202 and the order in which to record them. In the illustrative embodiment, adding means 206 is implemented with an Active X Control button labeled "Add". The user moves the mouse or cursor over the desired service in the list of available services 202 so that the selected track is highlighted on the user display, and then clicks the "Add" button. The selected track then appears in the list of selected services 204.

[0011] In the illustrative embodiment, user interface 200 includes removal means 208, implemented with an Active X Control button labeled "Remove", that allows the user to remove a selection by moving the mouse or cursor over the desired track in the list of selected services 204 such that the desired track becomes highlighted, and then clicking on "Remove" button. The highlighted track then disappears from the list of selected services 204 and is not included to be recorded when the CD is created. All selected tracks may be removed from the list of selected services 204 via a "Clear All" button 210.

[0012] User interface 200 also includes writing means 212 that causes the tracks listed in the list of selected services 204 to be written to the CD media. In the illustrative embodiment, writing means 212 is implemented as an Active X Control button labeled "Write Disc Now" that launches a write CD write operation. The write CD operation (not shown) handles the communication interface between the user's computer and the CD writer drive.

[0013] Preferably, user interface 200 includes scheduling means 214 to allow the user to schedule the tracks in the list of selected services 204 to be written to CD. In the illustrative embodiment, scheduling means 214 is an Active X Control button labeled "Schedule", which when clicked on, launches a scheduling application.

[0014] One application that the scheduling feature is particularly useful in is as a personal news recorder for making custom news and information discs. In this application, the user selects a number and order of available services desired such as the evening news, stock quotes, and entertainment news, then selects a desired ready time that the disc should be ready for pick-up, and activates the scheduling means 214 by clicking on the "Schedule" button. Scheduling means 214 launches a scheduling operation that schedules the download of the user-selected tracks and the time to launch the write operation for writing the tracks to the CD, such that the CD will be ready at the desired ready time. Thus, the CD is automatically created in the user-customized format without hands-on intervention and is ready for the user at his/her convenience. The user-customized profile (selected services, order, and desired ready time) may be scheduled to create a disc as often as the user desires.

[0015] User interface 200 may include additional features such as the ability to add or configure the list of available services. In the illustrative embodiment, this feature is provided by an Active X Control button 216 labeled "Add New Service", which launches a setup/configuration application when activated. Other features may include help facilities 218, information facilities 220 about the product or company, a link 222 to the provider's home web page, and more.

[0016] The illustrative embodiment of user interface 200 also provides previewing means 224, implemented as Active X Control button labeled "Preview". Previewing means 224 allows the user to highlight a track from the list of available services 202 and then activate the previewing means 224 to listen to or watch the highlighted audio/video track (or a portion of it) before deciding whether or not to add it to the list of selected services 204.

[0017] One application that this feature is particularly useful in is in a music retriever application. For example, a recording company may act as a service provider and supply a list of available audio tracks as the list of available services 202. Using the previewing means 224, the user may listen to a portion of a track in the list of available services 202 before adding it to the list of selected services which will be written to CD. Thus, the user can preview the track before writing it to CD in order to prevent unnecessary writing to the CD of those tracks ultimately not wanted.

[0018] FIG. 3 is a flowchart of the steps taken by a user to perform the method of the invention. Initially, the create CD application must be installed on the users system. This is performed either in a step 302 by navigating to a website that offers the create CD application for download and then downloading and installing the application on the users system in step 304. Alternatively, the application may be provided on an install CD that is provided by the CD drive manufacturer/supplier. In the preferred embodiment, the application is provided in a Microsoft format .CAB file, implemented as an ActiveX control that resides on an HTML web page of a web site. A cabinet is a single file created using Lempel-Ziv compression to hold a number of compressed files, and is used to save space and time during software distribution. During installation of a program, the compressed files in a cabinet are decompressed and copied to an appropriate directory for the user. In operation, the user navigates to the web site where the application is offered, and clicks on an HTML icon to download the create CD application. The cab file is then downloaded to the user's computer where it automatically unzips and decompresses itself and launches the user interface. As an alternative, in a Netscape Navigator environment, the application may be implemented as a plug-in.

[0019] Once the application is installed on the users system, the user launches the application as part of the local Internet browser in a step 308. This step may be performed automatically when the create CD application is installed, or may be performed manually by the user by clicking on an icon associated with the application on the desktop or within a program. Launching the application brings up the user interface of the application on a page (e.g., user interface 200 of FIG. 2) within the user's browser. Once the user interface is displayed, the user may then add new services in step 312. Adding new services is performed by activating a configuration application which registers the URL of an authorized multimedia service provider with the application. The authorization of an multimedia service provider is typically obtained via an authorization code provided by prior agreement between the multimedia service provider and application vendor.

[0020] The user selects services from the list of available services (or deselects services from the list of selected services) in step 310. The content of the available services is provided by the server side of the multimedia track providers' web site. In the illustrative embodiment, the format of the multimedia services content is a RealAudio® ".ra" file captured to a ".WAV" file. Service providers that wish to make their services available via the create CD application may be required to support a particular services content file format specified by the create CD application developer, or alternatively, the create CD application may include a file format converter that converts a number of file formats to the particular format that is written to the CD. Thus, for example, when a .ra audio/video track is downloaded from a service provider, the application may be implemented to convert the .ra track to a .wav file before writing the track to the CD. One embodiment may include a virtual sound card configured to convert audio/video tracks into a format that can be written to a CD.

[0021] The user then directs the application to write the disk immediately in step 314, or sets up the scheduler to schedule the write to the disk in step 316. In one implementation, the source (URL), time and/or size, data type, and other conversion or calculation parameters associated with a selected service are registered in a registration file which the write application reads when it is launched. If the disk is to be written immediately via step 314, the registration file is passed as a parameter to the write application. The write application downloads each selected track designated in the registration file and then writes each downloaded track to the CD 112 in the CD writer 110 via the communication interface 114. In the preferred embodiment, communication interface 114 is a SCSI interface communicating via calls to Adaptec Easy-CD Toolkit (XCD) library, version 2.0, manufactured by Adaptec, Inc. The programmer's references are detailed in Adaptec Easy-CD Toolkit (XCD) 2.0, Part #: 100023 - Manual revision: (G rev/19 Jan. 96), 2/20/97.

[0022] If the write to the CD is to be scheduled in step 316 for a later time, the user launches the scheduling application. Preferably, the scheduling application prompts the user for a desired completion time and is passed the registration file as a parameter. Then, scheduling application calculates how long the download and write to the CD will take based on the size, data type and sampling rates, modem speed, average network load for the requested time of day, and other system parameters or parameters contained the registration file to determine a time to launch the write application. Scheduling application then schedules the write application for that time. When the scheduled time arrives, the write application is launched, which downloads the selected tracks and writes them out to the CD writer drive in step 318.

[0023] Appendix A is an illustrative embodiment of an example user interface control file for a personal news recorder application of the create CD application of the invention. The user interface is implemented in Microsoft®'s Visual Basic. Section 1 of Appendix A defines each of the user interface boxes and command buttons, which include the main window of the personal news recorder, the operation status list, the preview timer, the preview command button, the help text box, the visit vendor picture box, the selected services list box, the available services list box, the write disc now command button, the schedule command button, the add new service command button, the clear all command button, the remove selected service command button, the add selected service command button, the help timer, the about this program command button, the help command button, the "available services" and "selected services" labels, and the borders and outlines. Section 2 of Appendix A contains the constant definitions and variable and external API declarations. For example, external API's for playing sound 'WINMM.DLL" and for writing a wave file to CD 'WRITE WAV.DLL" are declared in this section. Section 3 of Appendix A includes pseudocode for the subroutines associated with the user interface. These include routines for disabling all command buttons, enabling all command buttons, recording a track, responding to a click on the "About this Program" command button, adding a track to the selected services list in response to a click on the "Add" command button, responding to click on the "Add New Service" command button, clearing the entire selected services list in response to a click on the "Clear" command button, responding to a click on the "Help" command button, responding to a click on the "Preview" command button, responding to a click on the "Remove" command button, responding to a click on the "Schedule" command button, and writing to the CD in response to a click on the "Write CD Now" command button. Each of the subroutines may call external routines to accomplish its action. For example, the routine WriteDiscNow_Click() calls external API writewav(), a C implementation of the code used to write to the CD writer.

[0024] Appendix B is an illustrative embodiment of the C implementation for API writewav() which handles the writing out of tracks to the CD writer drive. Writewav() links in the Adaptec Easy-CD Toolkit (XCD) library, version 2.0, man-

ufactured by Adaptec, Inc., that allows it to communicate with the CD writer via a SCSI interface.

[0025] The implementation is an instance of the user interface window defined in Appendix A. The instance is created via an ActiveX control and displayed within the Internet browser. The ActiveX control handles all of the processing of user input mouse clicks through the procedures defined in the Visual Basic code in Appendix A.

[0026] Although the invention has been described in terms of the illustrative embodiments, it will be appreciated by those skilled in the art that various changes and modifications may be made to the illustrative embodiments without departing from the spirit or scope of the invention. It is intended that the scope of the invention not be limited in any way to the illustrative embodiment shown and described but that the invention be limited only by the claims appended hereto.

APPENDIX A

Section 1

```
Begin VB.UserControl PersonalNewsRecorder
   BackColor       =   &H00C0C0C0&
   ClientHeight    =   5895
   ClientLeft      =   0
   ClientTop       =   0
   ClientWidth     =   9120
   ForeColor       =   &H00000000&
   PropertyPages   =   "pnr.ctx":0000
   ScaleHeight     =   5895
   ScaleWidth      =   9120
   Begin VB.ListBox OperationStatusList
      BackColor          =   &H00000000&
      ForeColor          =   &H00FFFFFF&
      Height             =   4770
      Left               =   1800
      TabIndex           =   19
      Top                =   240
      Visible            =   0   'False
      Width              =   4935
   End
   Begin VB.Timer PreviewTimer
      Enabled            =   0   'False
      Interval           =   500
      Left               =   1320
      Top                =   0
   End
   Begin VB.CommandButton Preview
      BackColor          =   &H00C0C0C0&
      Caption            =   "Preview"
      Height             =   495
      Left               =   3240
      TabIndex           =   18
      ToolTipText        =   "Preview the currently highlighted selection
from the list of available services"
      Top                =   960
      Width              =   975
   End
   Begin VB.TextBox HelpText
      BackColor          =   &H00FF0000&
      ForeColor          =   &H00FFFFFF&
      Height             =   735
      Index              =   4
      Left               =   6000
      MultiLine          =   -1  'True
      TabIndex           =   16
      Text               =   "pnr.ctx":0004
      Top                =   4320
      Visible            =   0   'False
      Width              =   1815
   End
   Begin VB.PictureBox VisitHP
      Appearance         =   0  'Flat
      BackColor          =   &H80000005&
      ForeColor          =   &H80000008&
      Height             =   855
      Left               =   2400
```

```
            Picture         =   "pnr.ctx":004D
            ScaleHeight     =   825
            ScaleWidth      =   6465
            TabIndex        =   17
            ToolTipText     =   "Visit HP to find out more about cool new
products, including the HP Surestore CD-Writer 6020i"
            Top             =   4800
            Width           =   6495
        End
        Begin VB.TextBox HelpText
            BackColor       =   &H0000FFFF&
            ForeColor       =   &H00000000&
            Height          =   285
            Index           =   3
            Left            =   6360
            MultiLine       =   -1  'True
            TabIndex        =   15
            Text            =   "pnr.ctx":13C5F
            Top             =   3480
            Visible         =   0       'False
            Width           =   1575
        End
        Begin VB.TextBox HelpText
            BackColor       =   &H0000FFFF&
            ForeColor       =   &H00000000&
            Height          =   285
            Index           =   2
            Left            =   5040
            MultiLine       =   -1  'True
            TabIndex        =   14
            Text            =   "pnr.ctx":13C79
            Top             =   1920
            Visible         =   0       'False
            Width           =   1575
        End
    Begin VB.TextBox HelpText
            BackColor       =   &H0000FFFF&
            ForeColor       =   &H00000000&
            Height          =   285
            Index           =   1
            Left            =   360
            MultiLine       =   -1  'True
            TabIndex        =   13
            Text            =   "pnr.ctx":13C93
            Top             =   1320
            Visible         =   0       'False
            Width           =   2535
        End
        Begin VB.TextBox HelpText
            BackColor       =   &H0000FFFF&
            ForeColor       =   &H00000000&
            Height          =   495
            Index           =   0
            Left            =   2520
            MultiLine       =   -1  'True
            TabIndex        =   12
            Text            =   "pnr.ctx":13CB7
            Top             =   480
            Visible         =   0       'False
```

```
        Width           =   2895
    End
    Begin VB.ListBox SelectedList
        BackColor       =   &H00808080&
        BeginProperty Font
            Name        =   "MS Sans Serif"
            Size        =   8.25
            Charset     =   0
            Weight      =   700
            Underline   =   0       'False
            Italic      =   0       'False
            Strikethrough   =   0       'False
        EndProperty
        ForeColor       =   &H0000FFFF&
        Height          =   3795
        Left            =   4440
        TabIndex        =   11
        ToolTipText     =   "Click on a service to highlight it"
        Top             =   600
        Width           =   2775
    End
    Begin VB.ListBox AvailableList
        BackColor       =   &H00808080&
        BeginProperty Font
            Name        =   "MS Sans Serif"
            Size        =   8.25
            Charset     =   0
            Weight      =   700
            Underline   =   0       'False
            Italic      =   0       'False
            Strikethrough   =   0       'False
        EndProperty
        ForeColor       =   &H0000FFFF&
        Height          =   3795
        Left            =   240
        TabIndex        =   10
        ToolTipText     =   "Click on a service to highlight it"
        Top             =   600
        Width           =   2775
    End
    Begin VB.CommandButton WriteDiscNow
        BackColor       =   &H00C0C0C0&
        Caption         =   "Write Disc Now"
        DownPicture     =   "pnr.ctx":13CFC
        Height          =   735
        Left            =   7440
        Picture         =   "pnr.ctx":1413E
        Style           =   1  'Graphical
        TabIndex        =   7
        ToolTipText     =   "Write a disc now containing the selected
services"
        Top             =   3600
        Width           =   1455
    End
    Begin VB.CommandButton Schedule
        BackColor       =   &H00C0C0C0&
        Caption         =   "Schedule"
        DownPicture     =   "pnr.ctx":14580
        Height          =   735
```

```
            Left            =       7440
            Picture         =       "pnr.ctx":149C2
            Style           =       1  'Graphical
            TabIndex        =       6
            ToolTipText     =       "Schedule the download and disc write for
a later time"
            Top             =       2640
            Width           =       1455
        End
        Begin VB.CommandButton AddNewService
            BackColor       =       &H00C0C0C0&
            Caption         =       "Add New Service"
            Height          =       855
            Left            =       7440
            Picture         =       "pnr.ctx":14E04
            Style           =       1  'Graphical
            TabIndex        =       5
            ToolTipText     =       "Add a new service to the list of available
services"
            Top             =       840
            Width           =       1455
        End
        Begin VB.CommandButton Clear
            BackColor       =       &H00C0C0C0&
            Caption         =       "Clear All"
            Height          =       495
            Left            =       3240
            TabIndex        =       4
            ToolTipText     =       "Clear all entries from the list of selected
services"
            Top             =       3600
            Width           =       975
        End
        Begin VB.CommandButton Remove
            BackColor       =       &H00C0C0C0&
            Caption         =       "<- Remove"
            Height          =       495
            Left            =       3240
            TabIndex        =       3
            ToolTipText     =       "Remove the highlighted service from the
list of selected services"
            Top             =       2520
            Width           =       975
        End
        Begin VB.CommandButton Add
            BackColor       =       &H00C0C0C0&
            Caption         =       "Add ->"
            Height          =       495
            Left            =       3240
            TabIndex        =       2
            ToolTipText     =       "Add the highlighted service to the list of
selected services"
            Top             =       1920
            Width           =       975
        End
        Begin VB.Timer HelpTimer
            Enabled         =       0       'False
            Interval        =       3000
            Left            =       720
```

```
        Top             =   0
    End
    Begin VB.CommandButton About
        BackColor       =   &H00C0C0C0&
        Caption         =   "About"
        DownPicture     =   "pnr.ctx":155FE
        Height          =   855
        Left            =   1320
        Picture         =   "pnr.ctx":15A40
        Style           =   1  'Graphical
        TabIndex        =   1
        ToolTipText     =   "General information about this program"
        Top             =   4800
        Width           =   975
    End
    Begin VB.CommandButton Help
        BackColor       =   &H00C0C0C0&
        Caption         =   "Help"
        DownPicture     =   "pnr.ctx":15E82
        Height          =   855
        Left            =   240
        Picture         =   "pnr.ctx":162C4
        Style           =   1  'Graphical
        TabIndex        =   0
        ToolTipText     =   "Quick instructions on how to use this
program"
        Top             =   4800
        Width           =   975
    End
    Begin VB.Timer InitTimer
        Interval        =   1
        Left            =   120
        Top             =   0
    End
    Begin VB.Label AvailableServices
        BackColor       =   &H00C0C0C0&
        Caption         =   "Available Services"
        ForeColor       =   &H00000000&
        Height          =   255
        Left            =   960
        TabIndex        =   9
        Top             =   240
        Width           =   1455
    End
    Begin VB.Label SelectedServices
        BackColor       =   &H00C0C0C0&
        Caption         =   "Selected Services"
        ForeColor       =   &H00000000&
        Height          =   255
        Left            =   5160
        TabIndex        =   8
        Top             =   240
        Width           =   1335
    End
    Begin VB.Line Separator
        BorderColor     =   &H00000000&
        X1              =   240
        X2              =   8880
        Y1              =   4680
```

```
            Y2              =       4680
        End
        Begin VB.Shape Outline
            BackColor       =       &H00C0C0C0&
            BorderColor     =       &H00000000&
            Height          =       5655
            Left            =       120
            Top             =       120
            Width           =       8895
        End
End
```

Section 2

```
Attribute VB_Name = "PersonalNewsRecorder"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = True
' Debug flag
#Const DebugFlag = False

' Constants
Const MaxAvailable = 1000
Const MaxSelected = 99

Const SND_SYNC = &H0        ' For multimedia (sound) API
Const SND_ASYNC = &H1
Const SND_NODEFAULT = &H2
Const SND_LOOP = &H8
Const SND_NOSTOP = &H10

Const CD_OK = 0             ' For Ron's CD-R lib functions

Const HPWebSite = "http://www.hp.com/isgsupport/cdr/pi/index.html"

' Globals :-)
Dim CurrentIndex As Integer
Dim CurrentSound As String
Dim WelcomeSound As String
Dim HelpSound As String
Dim SilentSound As String
Dim Descriptor(MaxAvailable) As String
Dim FileName(MaxAvailable) As String
Dim CurrentTrack As String
Dim CloseFlag As Long

' External API declarations
Private Declare Function sndPlaySound Lib "WINMM.DLL" Alias _
    "sndPlaySoundA" (ByVal lpszSoundName As String, ByVal uFlags As _
    Long) As Long
Private Declare Function CD_Write Lib "WRITEWAV.DLL" Alias _
    "writewav" (ByVal lpszWaveName As String, ByVal CloseFlag As Long) _
    As Long
```

Section 3

```
' Disable all command buttons
Private Sub DisableButtons()
        Preview.Enabled = False
        Add.Enabled = False
```

```
        Remove.Enabled = False
        Clear.Enabled = False
        AddNewService.Enabled = False
        Schedule.Enabled = False
        WriteDiscNow.Enabled = False
        Help.Enabled = False
        About.Enabled = False
        VisitHP.Enabled = False
End Sub

' Enable all command buttons
Private Sub EnableButtons()
        Preview.Enabled = True
        Add.Enabled = True
        Remove.Enabled = True
        Clear.Enabled = True
        AddNewService.Enabled = True
        Schedule.Enabled = True
        WriteDiscNow.Enabled = True
        Help.Enabled = True
        About.Enabled = True
        VisitHP.Enabled = True
End Sub

' Start recording a track. This uses a blocking call.
Private Sub RecordTrack()
        #If DebugFlag = True Then
          MsgBox "RecordTrack: Starting write operation", vbInformation, "DEBUG"
        #End If
        x& = CD_Write(CurrentTrack, CloseFlag)
        #If DebugFlag = True Then
          MsgBox "RecordTrack: Write operation completed, retcode = " + Str(x&),
        vbInformation, "DEBUG"
        #End If
        On Error Resume Next
        Kill CurrentTrack
        On Error GoTo 0
        If x& <> CD_OK Then
          MsgBox "Bad news. CD Write failed (" + Str(x&) + ").", vbCritical, "Write error"
          OperationStatusList.Clear
          OperationStatusList.Visible = False
          SelectedList.Clear
          UserControl.MousePointer = vbDefault
          EnableButtons
        Else
          CurrentIndex = CurrentIndex + 1
          #If DebugFlag = True Then
            MsgBox "RecordTrack: CurrentIndex is " + Str(CurrentIndex), vbInformation,
        "DEBUG"
          #End If
          If CurrentIndex >= SelectedList.ListCount Then
            OperationStatusList.AddItem "Closing session"
            OperationStatusList.Refresh
            CurrentTrack = ""
            x& = CD_Write(CurrentTrack, CloseFlag)
            If x& <> CD_OK Then
              MsgBox "Bad news. CD Close Session failed (" + Str(x&) + ").", vbCritical,
        "Close session error"
            Else
```

```
            MsgBox "Disc successfully written", vbInformation, "Disc done"
        End If
        OperationStatusList.Clear
        OperationStatusList.Visible = False
        SelectedList.Clear
        UserControl.MousePointer = vbDefault
        EnableButtons
      Else
        OperationStatusList.AddItem "Downloading " +
Descriptor(SelectedList.ItemData(CurrentIndex))
        OperationStatusList.Refresh
        UserControl.AsyncRead FileName(SelectedList.ItemData(CurrentIndex)),
VbAsyncTypeFile, "RecordTrack" + Str(SelectedList.ItemData(CurrentIndex))
      End If
    End If
End Sub

' When the user clicks on About, we display a msgbox with
' info about the progam.
Private Sub About_Click()
    s$ = "Personal News Recorder" + Chr$(10) + Chr$(13) + Chr$(10) + Chr$(13)
    s$ = s$ + "This demonstration program presents an interface and working prototype
    for retrieval of news information directly from the web to a CD-Writer. "
    s$ = s$ + "The program is an ActiveX component delivered to your system from
    within the web browser environment."
    MsgBox s$, , "About the Personal News Recorder"
End Sub

' When the user clicks "add", add the item to the selected list.
Private Sub Add_Click()
    If AvailableList.ListIndex >= 0 Then
      If SelectedList.ListCount < MaxSelected Then
        SelectedList.AddItem Descriptor(AvailableList.ListIndex)
        SelectedList.ItemData(SelectedList.ListCount - 1) = AvailableList.ListIndex
      Else
        MsgBox "Sorry, you can only select up to " + Str(MaxSelected) + " services.",
    vbInformation, "Too many selected services"
      End If
    Else
      MsgBox "Highlight an available service, then click Add to add it to the list of
    selected services.", vbInformation, "Nothing highlighted"
    End If
End Sub

' Add new service
Private Sub AddNewService_Click()
    ' Register new service
    MsgBox "Allow the user to add new items to the list of available services.",
    vbInformation, "Add New Services"
End Sub

' When the user clicks "clear" clear the entire selected list.
Private Sub Clear_Click()
    If SelectedList.ListCount > 0 Then
      SelectedList.Clear
    Else
      MsgBox "There are no selected services to clear!", vbInformation, "No selected
    services"
    End If
```

```
End Sub

' When the user clicks Help, we place the first help pointers
' on the screen, then start the help autotimer.
Private Sub Help_Click()
        HelpText(0).Visible = True
        wFlags% = SND_ASYNC Or SND_NODEFAULT
        x% = sndPlaySound(HelpSound, wFlags%)
        HelpTimer.Enabled = True
End Sub

' When a Help is "in progress", each time the help timer ticks
' we set the next help queue card for the user to view.
Private Sub HelpTimer_Timer()
        If HelpText(0).Visible = True Then
           HelpText(0).Visible = False
           HelpText(1).Visible = True
           wFlags% = SND_ASYNC Or SND_NODEFAULT
           x% = sndPlaySound(HelpSound, wFlags%)
        ElseIf HelpText(1).Visible = True Then
           HelpText(1).Visible = False
           HelpText(2).Visible = True
           wFlags% = SND_ASYNC Or SND_NODEFAULT
           x% = sndPlaySound(HelpSound, wFlags%)
        ElseIf HelpText(2).Visible = True Then
           HelpText(2).Visible = False
           HelpText(3).Visible = True
           wFlags% = SND_ASYNC Or SND_NODEFAULT
           x% = sndPlaySound(HelpSound, wFlags%)
        ElseIf HelpText(3).Visible = True Then
           HelpText(3).Visible = False
           HelpText(4).Visible = True
           wFlags% = SND_ASYNC Or SND_NODEFAULT
           x% = sndPlaySound(HelpSound, wFlags%)
        ElseIf HelpText(4).Visible = True Then
           HelpText(4).Visible = False
           HelpTimer.Enabled = False
        End If
End Sub

' Initialization timer.  Reads in the .ini file asynchronously.
Private Sub InitTimer_Timer()
        InitTimer.Enabled = False
        DisableButtons
        UserControl.MousePointer = vbHourglass
        #If DebugFlag = True Then
           MsgBox "Reading ini file", , "DEBUG"
        #End If
        UserControl.AsyncRead "pnr.ini", VbAsyncTypeFile, "IniFile"
End Sub


' When the user clicks "preview" go get the currently selected
' sound file and play it.  If a sound is already playing, this
' button is labeled "stop", so stop the current sound by playing
' the "silent" wave.  Actually playing a short, empty wave is
' necessary to force the waveform device to free up the current
' open wave file :-)
Private Sub Preview_Click()
```

```
        If Preview.Caption = "Preview" Then
          If AvailableList.ListIndex >= 0 Then
            UserControl.MousePointer = vbHourglass
            UserControl.AsyncRead FileName(AvailableList.ListIndex), VbAsyncTypeFile,
        "PreviewSound"
          Else
            MsgBox "You must first highlight a service from the list of available services by
        clicking on it.  Then press Preview to hear a preview of the service.", , "Make a
        selection"
            Exit Sub
          End If
          Preview.Caption = "Stop"
          Preview.ToolTipText = "Stop the currently playing preview"
        Else
          PreviewTimer.Enabled = False
          wFlags% = SND_NODEFAULT
          x% = sndPlaySound(SilentSound, wFlags%)
          Preview.Caption = "Preview"
          Preview.ToolTipText = "Preview the currently highlighted selection from the list of
        available services"
          On Error Resume Next
          Kill CurrentSound
          On Error GoTo 0
        End If
End Sub

' This timer polls to see if the preview is finished playing.
' (An alternative way to do this would be to check the waveform
' device status, but couldn't seem to get that to work right from
' w/in Visual Basic.)
' If the preview is done, clean up by resetting the button to
' say "Preview" and erasing the current preview sound file.
Private Sub PreviewTimer_Timer()
  ' Try to play a silent sound with SND_NOSTOP flag set.  This
  ' will only succeed if the current sound is finished playing.
        wFlags% = SND_NODEFAULT Or SND_NOSTOP
        x% = sndPlaySound(SilentSound, wFlags%)

        ' Return code non-zero indicates the sound played OK, therefore
        ' we know that the preview must have been finished!
        If x% <> 0 Then
          PreviewTimer.Enabled = False
          Preview.Caption = "Preview"
          Preview.ToolTipText = "Preview the currently highlighted selection from the list
        of available services"
          On Error Resume Next
          Kill CurrentSound
          On Error GoTo 0
        End If
        End Sub

        ' When the user clicks "remove" we remove the selected item.
        Private Sub Remove_Click()
        If SelectedList.ListIndex >= 0 Then
          SelectedList.RemoveItem SelectedList.ListIndex
        Else
          MsgBox "You must first highlight a selected service, then click Remove.",
        vbInformation, "No selected service"
        End If
```

```
End Sub

' Schedule write to CD
Private Sub Schedule_Click()
        ' Call external scheduler routine.
        MsgBox "This feature is not implemented in this prototype. In a real product, this
        would allow the user to schedule a recording for some time later.", vbInformation,
        "Schedule"
End Sub

Private Sub UserControl_Terminate()
        On Error Resume Next
        Kill WelcomeSound
        Kill HelpSound
        Kill CurrentSound
        Kill SilentSound
        On Error GoTo 0
End Sub

' When the user clicks the "Visit HP" link, we navigate to
' the HP Web Site defined by our constant. If a sound is playing,
' we stop it first by playing the "silent" wave. If no sound
' was playing, that's OK anyway since the wave is short & silent!
Private Sub VisitHP_Click()
        wFlags% = SND_NODEFAULT
        x% = sndPlaySound(SilentSound, wFlags%)
        UserControl.Hyperlink.NavigateTo HPWebSite
End Sub

' All async reads generate an event here at completion.
' This more or less implements a crude state machine to
' drive completion of the sequence of async read events
' necessary to load a whole set of files for recording.
Private Sub UserControl_AsyncReadComplete(AsyncProp As AsyncProperty)

        ' State: IniFile
        '    Read in the IniFile. Should add error handling someday :-)
        If AsyncProp.PropertyName = "IniFile" Then
          #If DebugFlag = True Then
            MsgBox "State: IniFile", , "DEBUG"
          #End If
          Open AsyncProp.Value For Input As #1
          Input #1, WelcomeSound
          Input #1, HelpSound
          Input #1, SilentSound
          i = 0
          While EOF(1) <> True
            Input #1, Descriptor(i)
            AvailableList.AddItem Descriptor(i)
            Input #1, FileName(i)
            i = i + 1
          Wend
          Close #1
          Kill AsyncProp.Value
          UserControl.AsyncRead WelcomeSound, VbAsyncTypeFile, "WelcomeSound"

        ' State: WelcomeSound
        ElseIf Left$(AsyncProp.PropertyName, Len("WelcomeSound")) = "WelcomeSound"
        Then
```

```
    #If DebugFlag = True Then
      MsgBox "State: WelcomeSound", , "DEBUG"
    #End If
    WelcomeSound = AsyncProp.Value
    wFlags% = SND_ASYNC Or SND_NODEFAULT
    x% = sndPlaySound(WelcomeSound, wFlags%)
    UserControl.AsyncRead HelpSound, VbAsyncTypeFile, "HelpSound"

  ' State: HelpSound
  ElseIf Left$(AsyncProp.PropertyName, Len("HelpSound")) = "HelpSound" Then
    #If DebugFlag = True Then
      MsgBox "State: HelpSound", , "DEBUG"
    #End If
    HelpSound = AsyncProp.Value
    UserControl.AsyncRead SilentSound, VbAsyncTypeFile, "SilentSound"

  ' State: SilentSound
  ElseIf Left$(AsyncProp.PropertyName, Len("SilentSound")) = "SilentSound" Then
    #If DebugFlag = True Then
      MsgBox "State: SilentSound", , "DEBUG"
    #End If
    SilentSound = AsyncProp.Value
    EnableButtons
    UserControl.MousePointer = vbDefault

  ' State: PreviewSound
  ElseIf Left$(AsyncProp.PropertyName, Len("PreviewSound")) = "PreviewSound"
  Then
    #If DebugFlag = True Then
      MsgBox "State: PreviewSound", , "DEBUG"
    #End If
    UserControl.MousePointer = vbDefault
    CurrentSound = AsyncProp.Value
    wFlags% = SND_ASYNC Or SND_NODEFAULT
    x% = sndPlaySound(CurrentSound, wFlags%)
    PreviewTimer.Enabled = True

  ' State: RecordTrack
  '   Erase the tmp file for track we just recorded (null first time, no harm)
  '   Start recording track.  When done, we'll start the next download.
  ElseIf Left$(AsyncProp.PropertyName, Len("RecordTrack")) = "RecordTrack" Then
    #If DebugFlag = True Then
      MsgBox "State: RecordTrack", , "DEBUG"
    #End If
    On Error Resume Next
    Kill CurrentTrack
    On Error GoTo 0
    CurrentTrack = AsyncProp.Value
    i = Val(Right$(AsyncProp.PropertyName, Len(AsyncProp.PropertyName) -
  Len("RecordTrack")))
    OperationStatusList.AddItem "Writing " + Descriptor(i)
    OperationStatusList.Refresh
    RecordTrack

  ' State: Unknown
  '   State machine hurls
  Else
    MsgBox "Huh?  Unknown state (" + AsyncProp.PropertyName + ")" + " ...state
  machine hurls...", VbExclamation, "Really Bad Error"
```

```
        End If
End Sub

' When the user is ready to write a disc, first make sure there
' are selected services.  Then bring up the operation status display
' and let 'er rip!
Private Sub WriteDiscNow_Click()

    ' Make sure they selected some stuff
    If SelectedList.ListCount > 0 Then

      ' Disable buttons
      DisableButtons

      ' See if they want to close the disc when done.
      ' (Someday might want to add this to an options screen!)
      'If MsgBox("Do you want to close the disc when it's done writing?  If you answer
Yes, you'll be able to play the disc in any audio CD player (including a computer
CD-ROM) but you won't be able to add anything to the disc later.  If you answer No,
you'll only be able to play the disc on a computer CD-ROM, however you can add
more material to the CD later and close it at that time.", vbInformation + vbYesNo,
"Close disc?") = vbYes Then
        ' CloseFlag = 1
      'Else
        ' CloseFlag = 0
      'End If
      CloseFlag = 0

      ' Make sure they want to go ahead!
      If MsgBox("Please make sure you have recordable media inserted in the drive.
Then press OK to proceed." + Chr$(10) + Chr$(13) + Chr$(10) + Chr$(13) + "(In a
real product, we could give an option here for printing jewel case artwork.)",
vbInformation + vbOKCancel, "Prepare media and confirm") = vbCancel Then
        EnableButtons
        Exit Sub
      End If

      ' Make the operation status box & cancel button visible
      OperationStatusList.Visible = True

      ' Begin download of files from selected list.  For now,
      ' we have single threaded state machine engine which just
      ' downloads a file, then records it, then repeats for the
      ' next file.  Right here, we just kick it off by starting the
      ' download of the first file.
      CurrentIndex = 0
      OperationStatusList.AddItem "Downloading " +
Descriptor(SelectedList.ItemData(CurrentIndex))
      OperationStatusList.Refresh
      UserControl.MousePointer = vbHourglass
      UserControl.AsyncRead FileName(SelectedList.ItemData(CurrentIndex)),
VbAsyncTypeFile, "RecordTrack" + Str(SelectedList.ItemData(CurrentIndex))

    ' Send 'em to help if nothing selected.
    Else
      MsgBox "You must first selected services that you want to record!  Press Help for
more information.", , "Select services"
      Exit Sub
    End If
```

End Sub

APPENDIX B

```
#include <windows.h>

#include "xcd.h"

BOOL WINAPI DllMain (HANDLE hModule, DWORD fdwReason, LPVOID lpReserved)
{
  switch (fdwReason)
  {
    case DLL_PROCESS_ATTACH:
      /* Code from LibMain inserted here.  Return TRUE to keep the
        DLL loaded or return FALSE to fail loading the DLL.

        You may have to modify the code in your original LibMain to
        account for the fact that it may be called more than once.
        You will get one DLL_PROCESS_ATTACH for each process that
        loads the DLL. This is different from LibMain which gets
        called only once when the DLL is loaded. The only time this
        is critical is when you are using shared data sections.
        If you are using shared data sections for statically
        allocated data, you will need to be careful to initialize it
        only once. Check your code carefully.

        Certain one-time initializations may now need to be done for
        each process that attaches. You may also not need code from
        your original LibMain because the operating system may now
        be doing it for you.
      */
      break;

    case DLL_THREAD_ATTACH:
      /* Called each time a thread is created in a process that has
        already loaded (attached to) this DLL. Does not get called
        for each thread that exists in the process before it loaded
        the DLL.

        Do thread-specific initialization here.
      */
      break;

    case DLL_THREAD_DETACH:
      /* Same as above, but called when a thread in the process
        exits.

        Do thread-specific cleanup here.
      */
      break;

    case DLL_PROCESS_DETACH:
      /* Code from _WEP inserted here.  This code may (like the
        LibMain) not be necessary.  Check to make certain that the
        operating system is not doing it for you.
      */
      break;
  }

  /* The return value is only used for DLL_PROCESS_ATTACH; all other
    conditions are ignored.  */
  return TRUE;  // successful DLL_PROCESS_ATTACH
```

```
}

//__declspec( dllexport )

__declspec (dllexport) DWORD __stdcall writewav(LPCSTR filename, long closeflag) {

        HXCD                hXcd;
        UINT32              u32Error,u32Err;
        DRVTABLE            DriveTable[16];
        INT32               i32DriveCount = 16;
        RECORDTRACK             RecordTrack;
        int                     driveselect;
        BOOLEAN                 fCloseDisc;

        u32Error = XcdNewInstance(&hXcd); // u32Error should == XCD_NOERR which is
= 0
        if(u32Error != XCD_NOERR) {
                u32Err = XcdDeselectDrv(hXcd);              // Cleanup
                u32Err = XcdDeleteInstance(hXcd);   // Cleanup
                return u32Error;
        };

        u32Error = XcdScanDrv(hXcd, DriveTable, &i32DriveCount);
        if(u32Error != XCD_NOERR) {
                u32Err = XcdDeselectDrv(hXcd);              // Cleanup
                u32Err = XcdDeleteInstance(hXcd);   // Cleanup
                return u32Error;
        };
        if (i32DriveCount == 0) { return XCD_UNEXPECTED-1; };

        driveselect = 0;
//      while(strncmp((DriveTable[driveselect]).szDescription, "HP", 2)) {
//              driveselect++;
//              if (driveselect >= i32DriveCount) { return XCD_UNEXPECTED;};
//      }

        u32Error = XcdSelectDrv(hXcd, &DriveTable[driveselect]); // Select the first cd-
writer
        if(u32Error != XCD_NOERR) {
                u32Err = XcdDeselectDrv(hXcd);              // Cleanup
                u32Err = XcdDeleteInstance(hXcd);   // Cleanup
                return u32Error;
        };

        u32Error = XcdTestUnitReady(hXcd);
        if(u32Error != XCD_NOERR) {
                u32Err = XcdDeselectDrv(hXcd);              // Cleanup
                u32Err = XcdDeleteInstance(hXcd);   // Cleanup
                return u32Error;
        };

        memset(&RecordTrack, 0, sizeof(RECORDTRACK));

        RecordTrack.fOnTheFly = FALSE;                  // Not using a VCD
        RecordTrack.u32Type = XCD_WAVE;                 // PCM Wave Audio file
format
        RecordTrack.u32Action = XCD_RECWRITE;   // Perform a speed test only
(XCD_RECTEST)
        RecordTrack.fDiscAtOnce = FALSE;            // Want TRACK AT ONCE
```

```
        RecordTrack.u32SessionType = XCD_CLOSECDDA;    // Only audio tracks
        RecordTrack.u32Speed = XCD_SPEED2X;            // select speed (2 times)

        fCloseDisc = FALSE;
        if(closeflag)
                fCloseDisc = TRUE;

        if (strlen(filename) == 0) {
                u32Error = XcdCloseSession (hXcd, XCD_CLOSECDDA, fCloseDisc ,
NULL);
                if(u32Error != XCD_NOERR) {
                        u32Err = XcdDeselectDrv(hXcd);          // Cleanup
                        u32Err = XcdDeleteInstance(hXcd);       // Cleanup
                        return u32Error;
                };
        } else {
                strcpy(RecordTrack.szTrackFileName,filename);
                u32Error = XcdRecord(hXcd, &RecordTrack, NULL); // Do it!
                if(u32Error != XCD_NOERR) {
                        u32Err = XcdDeselectDrv(hXcd);          // Cleanup
                        u32Err = XcdDeleteInstance(hXcd);       // Cleanup
                        return u32Error;
                };
        }

        u32Error = XcdDeselectDrv(hXcd);               // Cleanup
        if(u32Error != XCD_NOERR) {
                return u32Error;
        };

        u32Error = XcdDeleteInstance(hXcd);            // Cleanup
        if(u32Error != XCD_NOERR) {
                return u32Error;
        };

        return (u32Error);
}
```

Claims

1. A method for automatically downloading from the Internet and recording user-selected audio and/or visual tracks (204) onto a portable CD (112) via a web browser (104), said method comprising the steps of:

   determining a pre-selected multimedia track, said pre-selected multimedia track comprising a multi-media track available from a web site;
   downloading (318) said pre-selected multimedia track from said web site;
   recording (318) said downloaded multimedia track onto said portable CD.

2. A method in accordance with claim 1, comprising:

   selecting (310) said pre-selected multimedia track from a plurality of multimedia tracks (202) available over said Internet.

3. A method in accordance with claim 1 or 2, comprising:

navigating (302) to said web site using said web browser (104);
downloading (304) a compact disc (CD) creation application (106) from said web site, said CD creation application (106) comprising means (206) for selecting said pre-selected multimedia track (204) from a plurality of multimedia tracks (202) available over said Internet; and
launching (308) said CD creation application (106) from said web browser (104).

4. A method in accordance with claim 1, 2 or 3, comprising:

scheduling (316) said downloading step and said recording step to complete said recording step by a pre-specified time.

5. A method in accordance with claim 1, 2 or 3, comprising:

scheduling (316) said downloading step and said recording step to occur at a pre-specified time.

6. A method in accordance with claim 1, 2 or 3, comprising:

scheduling (316) said downloading step to occur at a pre-specified time.

7. A method in accordance with claim 1, 2 or 3, comprising:

scheduling (316) said recording step to occur at a pre-specified time.

8. A user interface for automatically downloading from the Internet and recording user-selected audio/visual tracks onto a portable CD via a web browser, comprising:

a list (202) of available multimedia tracks available for download from the Internet;
selection means (206) for selecting one of said available multimedia tracks from said list;
recording means (212, 214) for causing said selected one of said available multimedia tracks to be downloaded from the Internet and recorded onto a compact disc (CD) (112).

9. A user interface in accordance with claim 8, wherein:

said recording means comprises scheduling means (214) for scheduling said download of said selected one of said available multimedia tracks from the Internet and said recording onto said CD (112) to occur at a user-selectable time.

10. A user interface in accordance with claim 8, wherein:

said recording means comprises scheduling means (214) for scheduling said download of said selected one of said available multimedia tracks from the Internet and said recording onto said CD (112) to occur by a user-selectable time.

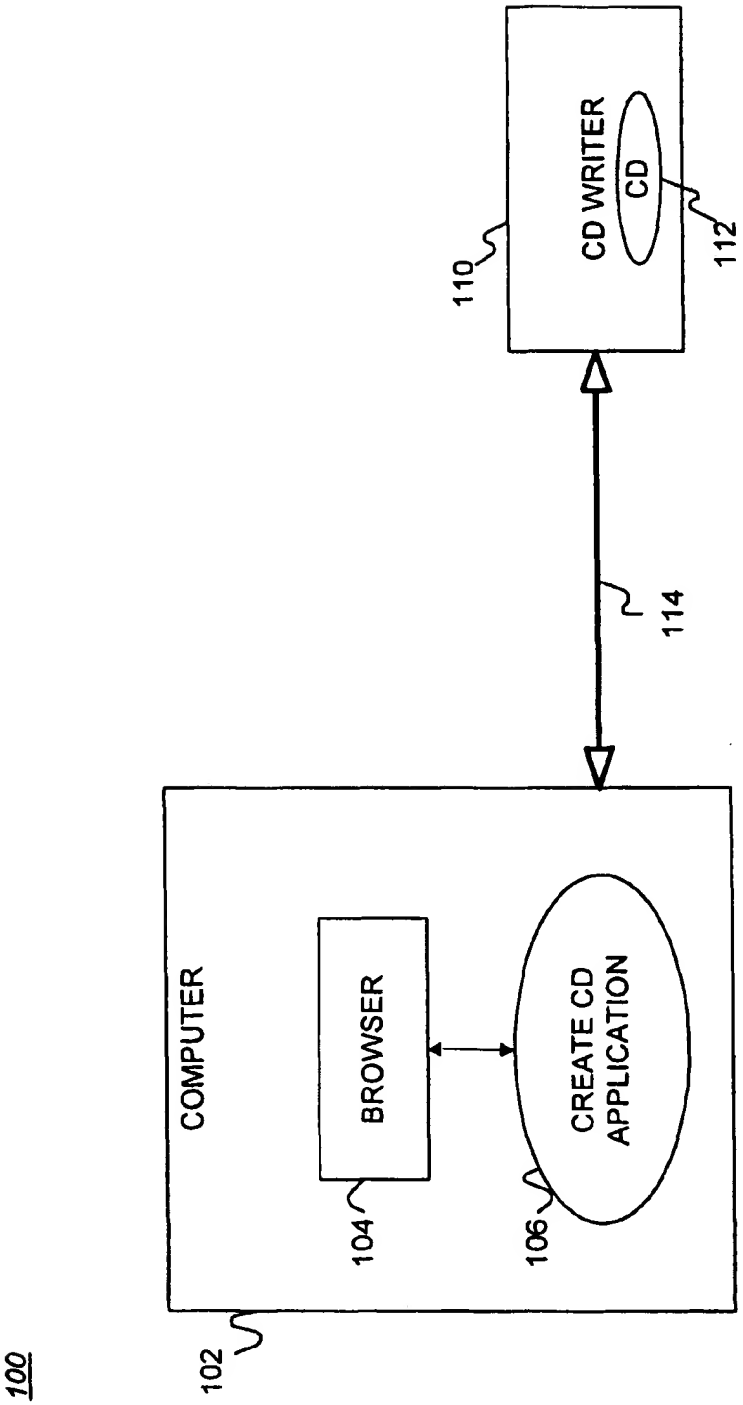
100
102
COMPUTER
BROWSER
104
CREATE CD APPLICATION
106
114
110
CD WRITER
CD
112

FIG. 1

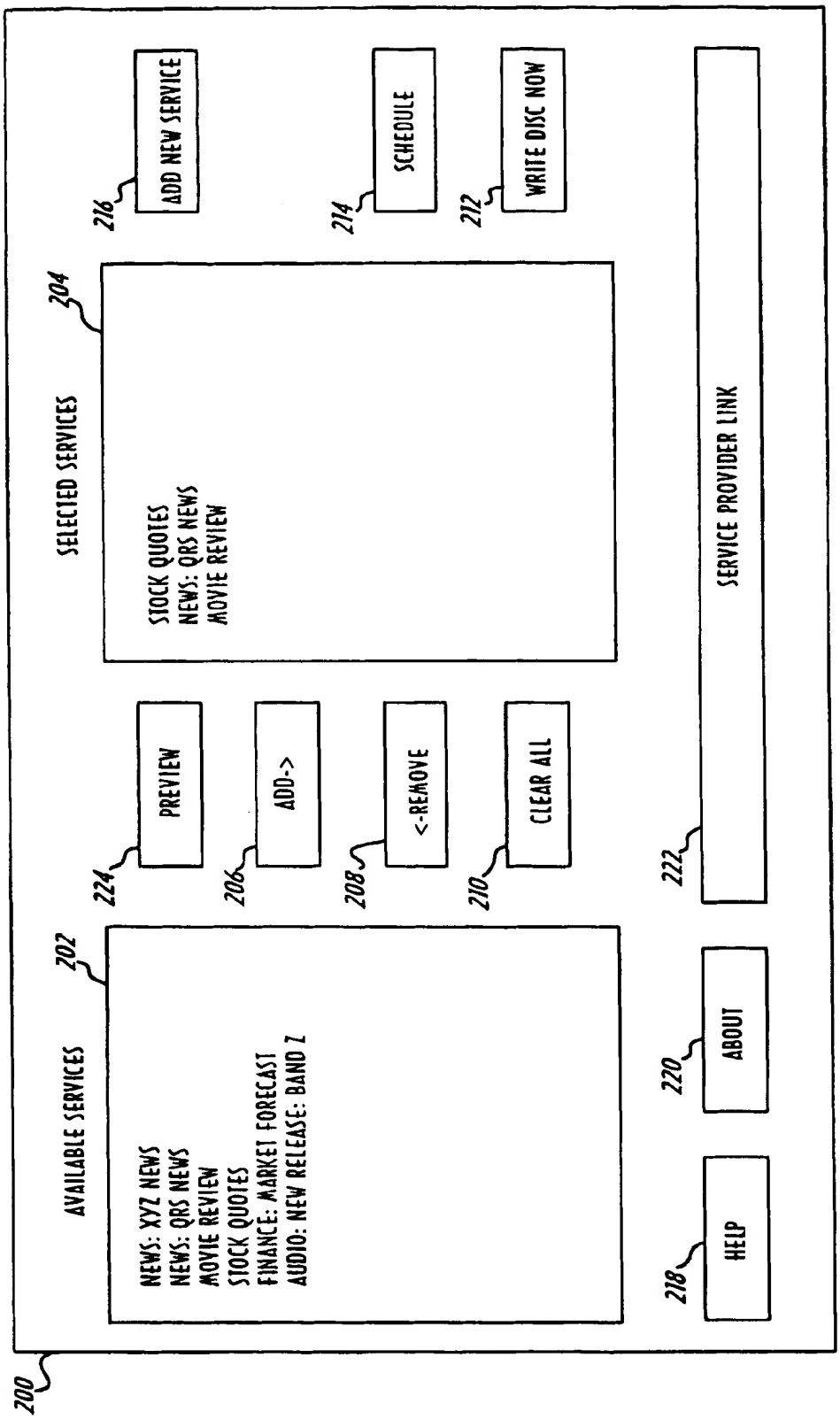
200
AVAILABLE SERVICES
202
NEWS: XYZ NEWS
NEWS: QRS NEWS
MOVIE REVIEW
STOCK QUOTES
FINANCE: MARKET FORECAST
AUDIO: NEW RELEASE: BAND Z
224
PREVIEW
206
ADD->
208
<-REMOVE
210
CLEAR ALL
SELECTED SERVICES
204
STOCK QUOTES
NEWS: QRS NEWS
MOVIE REVIEW
216
ADD NEW SERVICE
214
SCHEDULE
212
WRITE DISC NOW
222
SERVICE PROVIDER LINK
220
ABOUT
218
HELP

FIG. 2

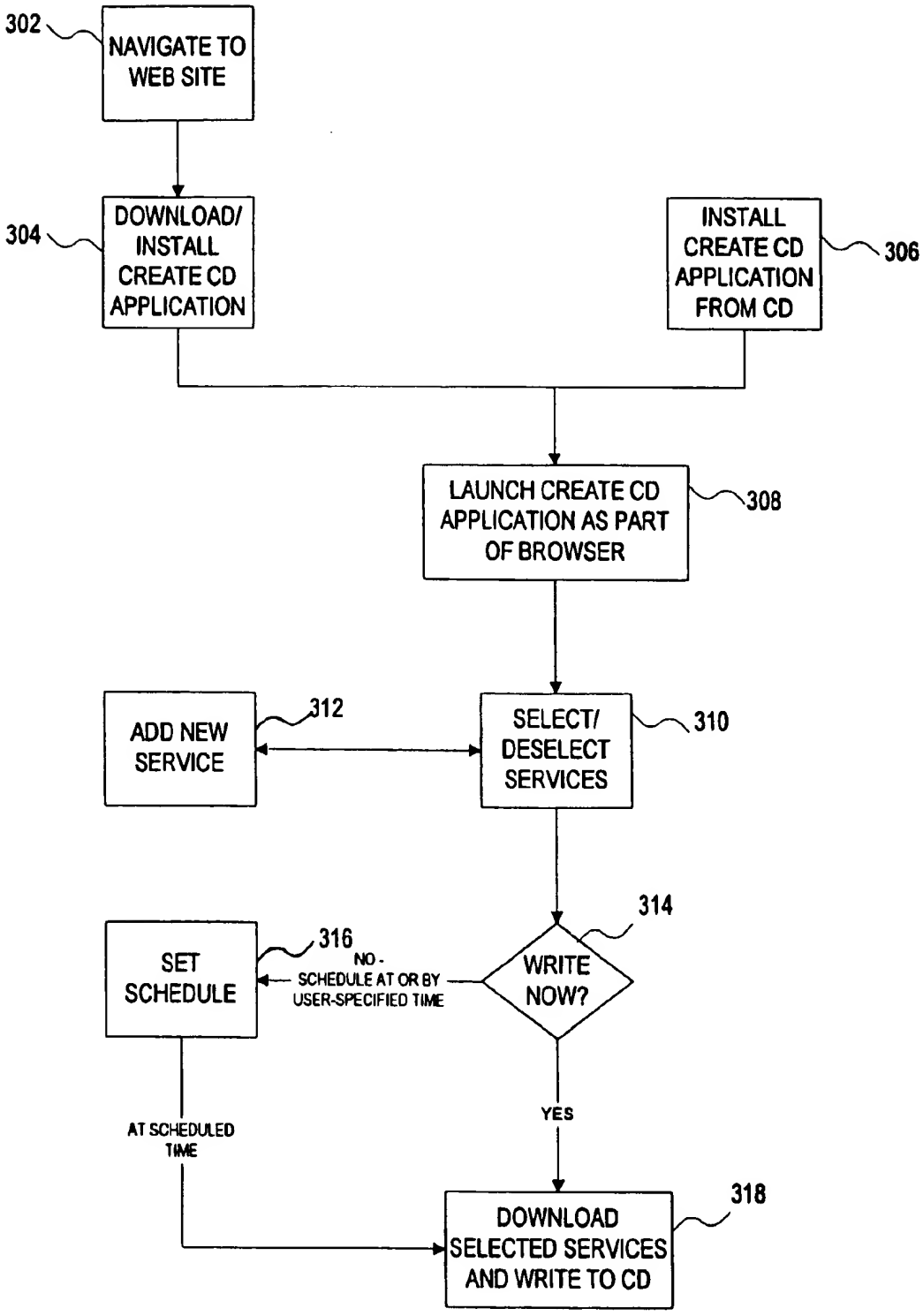
302
NAVIGATE TO WEB SITE
304
DOWNLOAD/ INSTALL CREATE CD APPLICATION
INSTALL CREATE CD APPLICATION FROM CD
306
LAUNCH CREATE CD APPLICATION AS PART OF BROWSER
308
ADD NEW SERVICE
312
SELECT/ DESELECT SERVICES
310
314
SET SCHEDULE
316
NO - SCHEDULE AT OR BY USER-SPECIFIED TIME
WRITE NOW?
AT SCHEDULED TIME
YES
DOWNLOAD SELECTED SERVICES AND WRITE TO CD
318

FIG. 3